Demo: An Interoperability Development and Performance Diagnosis Environment

JeongGil Ko
Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218, USA
jgko, terzis@cs.jhu.edu

Joakim Eriksson
Swedish Institute of Computer Science (SICS)
Box 1263, SE-16429 Kista, Sweden
joakime@sics.se

Nicolas Tsiftes
Swedish Institute of Computer Science (SICS)
Box 1263, SE-16429 Kista, Sweden
nvt@sics.se

Stephen Dawson-Haggerty
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720, USA
stevedh@eecs.berkeley.edu

Jean-Philippe Vasseur
Cisco Systems
11, Rue Camille Desmoulins, Issy Les Moulineaux, 92782, France
jpv@cisco.com

Mathilde Durvy
Cisco Systems
11, Rue Camille Desmoulins, Issy Les Moulineaux, 92782, France
mdurvy@cisco.com

Abstract

Interoperability is key to widespread adoption of sensor network technology, but interoperable systems have traditionally been difficult to develop and test. We demonstrate an interoperable system development and performance diagnosis environment in which different systems, different software, and different hardware can be simulated in a single network configuration. This allows both development, verification, and performance diagnosis of interoperable systems. Estimating the performance is important since even when systems interoperate, the performance can be sub-optimal, as shown in our companion paper that has been conditionally accepted for SenSys 2011.

Categories and Subject Descriptors

C.2.6 [**Computer-Communication Networks**]: Internetworking—*Standards*; D.2.12 [**Software**]: Interoperability; D.4.8 [**Software**]: Performance—*Simulation*

General Terms

Experimentation, Performance, Measurement

Keywords

IETF, IPv6, 6LoWPAN, RPL, Interoperability, Sensor Network, TinyOS, Contiki, Cooja

1 Interoperability Development and Performance Diagnosis

The Internet Protocol (IP), which has proved its interoperability and extensibility in the global Internet, is seen by many as a promising solution to the interoperability problem in low-power and lossy networks (LLNs) [1, 3, 6]. The Internet Engineering Task Force (IETF) has recently specified a number of protocols and adaptation layers that allow IPv6 to run over IEEE 802.15.4 link layers. The 6LoWPAN working group specified header compression and fragmentation for IPv6 over IEEE 802.15.4 and the IETF RoLL working group designed the RPL protocol as a proposed standard for IPv6 routing in LLNs.

Industrial standard practices with IPSO Alliance bake-off events or conformance testing against reference implementations are essential to attaining interoperable systems, but do not address performance problems that may arise [5]. The ability to run large-scale networks and to have a fully controlled environment is important for performance diagnosis. Network simulation provides both.

Our interoperability performance diagnosis environment consists of the Contiki simulation environment. The Contiki simulation environment consists of the Cooja network simulator, which provides bit-level network simulation of abstract motes and the MSPsim Tmote Sky emulator. The motes can be implemented in either Java, C, or even emulated versions of hardware motes. The MSPsim emulator provides cycle accurate emulation of the MSP430 and bit-level accurate emulation of the CC2420 radio transceiver.

The interoperability development and performance diagnosis framework complements but does not replace interoperability testing events. The interoperability diagnosis framework can be used both for the initial functional testing and diagnosis of performance problems. But to capture the effects of real wireless channel environments, physical interoperability testing events or testbed validation of the simulation results are essential.

2 Different Software: Contiki and TinyOS

Contiki and TinyOS both have their own, non-interoperable, protocol frameworks to allow for research and development without the constraints of a standardized framework. Contiki has long used the IP protocol framework [1, 2] but since there were no standards for IP routing and transmit-

Copyright is held by the author/owner(s).
SenSys'11, November 1–4, 2011, Seattle, WA, USA.
ACM 978-1-4503-0718-5/11/11

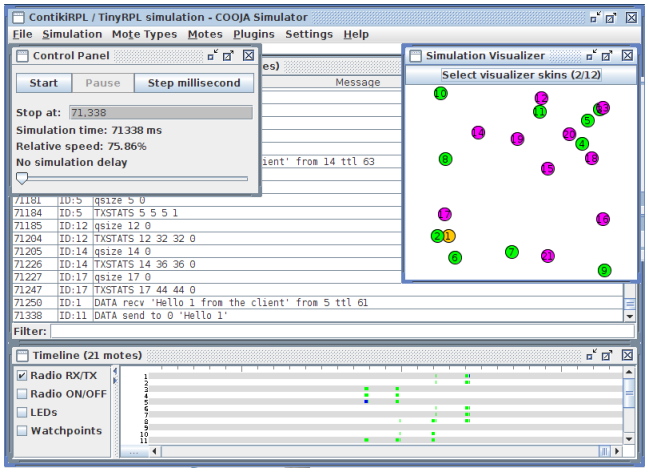

Figure 1. The Contiki simulation environment with Contiki Tmote Sky nodes (green and yellow) and TinyOS nodes (purple). Serial output from all nodes is shown in the Log listener window. The TimeLine pane at the bottom shows radio events from both Contiki and TinyOS motes.

ting IP packets, achieving interoperability was impossible.

With the introduction of the new IETF standards, both TinyOS and Contiki now support IP [1, 2, 4, 5]. Both stacks implement the IPv6 protocol, including ICMPv6, with support for the UDP and TCP transport protocols. Underneath the IPv6 layer, both software stacks implement the 6LoWPAN header compression and fragmentation layer. For routing, both Contiki and TinyOS implement the IETF RPL routing protocol as ContikiRPL and TinyRPL. Both RPL implementations support routing policies such as OF0 and MRHOF based on the ETX metric.

In the interoperability testing and performance diagnosis environment, motes with different software can be freely mixed. The simulator does not care whether it simulates Contiki or TinyOS systems, but treats them as black boxes. Nodes run the exact same binary images as the real hardware does. This allows the interoperability environment to run any operating system, both open source and proprietary.

3 Different Hardware: Tmote Sky and MicaZ

Interoperability is both about software and hardware. To allow cross-platform interoperability testing, we leverage the Contiki simulation environments platform independence feature where different hardware can be mixed in the same simulation. Specifically, in additional to the MSPsim Tmote Sky emulator, the Contiki simulation environment includes the AvroraZ MicaZ mote emulator. The Cooja simulator treats both as black boxes, which means that MicaZ and Tmote Sky motes can be freely mixed in the same network configuration. This makes it possible for us to test interoperability between Tmote Sky and MicaZ motes.

The Tmote Sky and the MicaZ both use the same IEEE 802.15.4 radio standard; therefore, interoperability at the physical and data link layers are possible. The simulation framework also supports other radio layers, such as the one used by the TR1001 radio, but since they do not interoperate with the 802.15.4 they cannot be ran in conjunction.

4 Next Step: Low-power Interoperable Systems

Interoperability has thus far been demonstrated without considering power efficiency. To attain low-power operation, radios must be duty cycled. Even though duty cycling mechanisms are being standardized through the IEEE 802.15.4e effort, interoperable implementations have yet to be demonstrated. Developing interoperable duty cycling mechanisms is difficult due to the precise timing requirements needed. Software with such timing requirements typically needs extensive hardware support for development, such as oscilloscopes. In the Contiki simulation environment, the TimeLine view makes it possible to inspect the behavior of duty cycling protocols at the microsecond level in a controlled environment. This ability allows us to achieve the next goal: low-power interoperability.

5 Demo Setup

As part of our demonstration we present a Contiki Simulation Environment with TinyOS and Contiki nodes. Figure 1 presents an example network configuration. We demonstrate how simulations are set up, how performance problems can be detected and debugged, and how timing-sensitive message exchanges can be studied in detail with the TimeLine view. For this portion of the demonstration we emphasize on how a simulation environments can be well-suited for sensor network interoperability performance testing.

To demonstrate interoperability outside of the simulator, we also show a small testbed environment where TinyOS and Contiki Tmote Sky nodes coexist to cooperatively exchange messages with an existing IPv6 infrastructure.

6 Additional Authors

Additional Authors: Andreas Terzis (Department of Computer Science, Johns Hopkins University, email: terzis@cs.jhu.edu) and Adam Dunkels (Swedish Institute of Computer Science (SICS), email: adam@sics.se) and David Culler (Computer Science Division, University of California, Berkeley, email:culler@eecs.berkeley.edu).

7 References

[1] A. Dunkels. Full TCP/IP for 8-bit architectures. In *Proceedings of The International Conference on Mobile Systems, Applications, and Services (MobiSys)*, San Francisco, CA, USA, May 2003.

[2] M. Durvy, J. Abeillé, P. Wetterwald, C. O'Flynn, B. Leverett, E. Gnoske, M. Vidales, G. Mulligan, N. Tsiftes, N. Finne, and A. Dunkels. Making Sensor Networks IPv6 Ready. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Raleigh, North Carolina, USA, November 2008.

[3] J. Hui and D. Culler. IP is Dead, Long Live IP for Wireless Sensor Networks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, Raleigh, North Carolina, USA, November 2008.

[4] J. Ko, S. Dawson-Haggerty, J. Hui, P. Levis, D. Culler, and A. Terzis. Connecting low-power and lossy networks to the internet. *IEEE Communications Magazine*, 49, April 2011.

[5] J. Ko, J. Eriksson, N. Tsiftes, S. Dawson-Haggerty, M. Durvy, J. Vasseur, A. Terzis, A. Dunkels, and D. Culler. Beyond Interoperability: Pushing the Performance of Sensornet IP Stacks. In *Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys)*, November 2011.

[6] J.P. Vasseur and A. Dunkels. *Interconnecting Smart Objects with IP: The Next Internet*. Morgan Kaufmann, 2010.